denver apple pi Seed

Volume 6 No.5 MAY 84

NEWSLETTER OF THE DENVER APPLE PI COMPUTER USERS' CLUB

Don't be Out of Sorts

by Steven Martin

In a recent article in the SEED Andre' van Meulebrouck discussed a modification to the bubble sort that gave an increase in performance. As I have always been fascinated by sorting algorithms I immediately tried Andre's routine and was pleased to see how well it worked. I also saw that Andre' might have missed a subtle point that improves the speed of this sort even further. Before I describe the improvement, however, I would like to digress a bit to a general discussion of sorting routines and how they work.

Bubble Sort Revisited...

The bubble sort is the simplest sorting method I am aware of and uses the least amount of coding of any sort. The basic bubble sort looks like this:

```
10  FOR I = 1 TO NUMOFCELLS
20  FOR L = 1 TO NUMOFCELLS - 1
30  IF  D(L) > D(L+1)  THEN
      T = D(L) :
      D(L) = D(L+1) :
      D(L+1) = T
40  NEXT L
50  NEXT I
100 END
```

Listing 1. Bubble sort.

Where D() is the Array we want to sort and NUMOFCELLS is the number of elements in D() to be sorted (i.e. DIM D(NUMOFCELLS)). How it works is simple. We go through the list as many times as there are elements in the array (line 10). We compare the elements that are next to each other (L and L+1) and if they are out of order we swap them (Line 30). We will check NUMOFCELLS items NUMOFCELLS times. In other words if we are checking 10 items we will make 100 comparisons (10*10=100), but if we are checking 1000 items we make 1 million comparisons! We can see that this will take even very fast computers a long time to sort big lists.

Note that on each pass through the array we have moved the smaller elements towards the top and the larger ones down. This is what gives the bubble sort its name. What if the list is already sorted? Or if the list is sorted after making only a few passes? The bubble sort keeps right on going until all of the passes are made. This is, obviously, innefficient and we can make a check that tells us if the list is sorted after any pass. This is a bubble sort with flag:

```
10  FOR I = 1 TO NUMOFCELLS
15  FLAG = 1
20  FOR L = 1 TO NUMOFCELLS - 1
30  IF  D(L) > D(L+1)  THEN
      T = D(L) :
      D(L) = D(L+1) :
      D(L+1) = T :
      FLAG = 0
40  NEXT L
50  IF FLAG = 1 THEN I = NUMOFCELLS
60  NEXT I
100 END
```

Listing 2. Bubble sort with flag.

FLAG is equal to 1 if the array is sorted and 0 if it is not. We first assume that it is sorted (Line 15). If in going through the list we have to make a change then the array was not sorted so we set FLAG to 0 (Line 30). Finally we check FLAG (Line 50) and if we are not sorted we repeat the process. If the list is in completely reversed order the flag will not speed up the bubble sort.

Here is where Andre's idea comes in. If we take a look at the swap routine (Line 30) we note that a large element is moved to the bottom and then, after the NEXT L command is performed this same element is again compared to the next item down. Thus the largest item will always end up at the bottom after the first pass, the second largest after the second pass and so forth. This is why Andre' prefers to call it a sink sort. After the first pass we no longer need to check the bottom element as their will be nothing larger than it. So we can save on comparison with each succesive pass. Andre's sort looks like this:

denver apple pi Seed

APPLE PI
P.O. BOX 17467
DENVER, COLORADO 80217

OFFICERS

PRESIDENT
Dave Eisler . . . . . (303) 694-6217
VICE-PRESIDENT
Bill Dowall . . . . . (303) 469-5638
TREASURER
Bob Hein . . . . . (303) 424-8357
TREASURE ASSISTANT
Rich Waite . . . . . (303) 451-9215
SECRETARY
Dave Nelson . . . . . (303) 238-4014
LIBRARIAN
Trent Hein . . . . . (303) 424-8357
LIBRARY ASSISTANTS
Dan Bentz . . . . . (303) 831-6052
Bob Gaiser . . . . . (303) 779-4033
Jim Lyons . . . . . (303) 469-2741

OFFICERS-AT-LARGE
Ron Heath . . . . . (303) 238-3019
Dennis Lunt . . . . . (303) 823-6200

SEED EDITOR
Randall Ansley . . . . . (303) 428-5627

ADVERTISING RATES

$20.00 . . . . . ¼ Page
$35.00 . . . . . ½ Page
$60.00 . . . . . Full Page

CIRCULATION OVER 600

The word **APPLE, APPLE COMPUTER** and the **APPLE LOGO** are registered trademarks of the Apple Computer Company Inc.

Representation of these trademarks in the 'Seed' does not constitute any expression of agreement or disagreement by Apple Computer Inc. with any articles or printed material found in the **SEED.** The opinions, policies and comments contained in the **'SEED'** are soley the responsibility of the Apple Pi users group officers and member.

GENERAL MEETING . . . 3RD

SWAP MEET . . . 6TH

BOARD MEETING . . . 9TH

MEMBERSHIP INFORMATION
$10 APPLICATION FEE AND $18 PER YEAR DUES

APPLE PI is an Apple Computer Users Group - not affiliated with Apple Computer, Inc. - that provides many member benefits, some of which are:

MONTHLY MEETINGS-

On the first Thursday of each month at 7:00 p.m. APPLE PI holds its general meeting at the Greene Center, Colorado School of Mines Campus, Golden, Colorado. This meeting is a combination of an informational meeting, new product announcements, and a short business meeting. The members are given an opportunity to buy computer supplies a little above cost. Software products are given away as door prizes.

On the Sunday following the general meeting there is an informal meet which is usually held at the home of a member, the location of which is announced in the newsletter and at the general meeting. This meeting is from 10:00 a.m. until 3:00 p.m. and provides a more informal environment to seek or give advice to fellow members and get copies of programs from the APPLE PI Library. Both of these meetings are open to the general public but only members have access to products that are sold.

PROGRAM LIBRARY-

APPLE PI currently has a library of 8,000 programs. All current disks are in 16 sector (3.3, Pascal, CP/M) format and are available to any member. APPLE PI Members can obtain copies from the library:

1) By coming to the monthly informal meeting (swap meet);
2) By written request to the club librarian (at any meeting or by mail);
3) By electronic mail on APPLE PI's bulletin board system:

When requesting copies, please include the following information:

a) Your Name, Address, and Phone number;
b) Disk Category and Volume Number;
c) Blank Diskette(s) or request to utilize a new disk (member will be charged $2.50 for the new disk);
d) Copy fee ($1.00 per diskette side unless you donate a program you have written at the same time and fill out a Progam Submission Form). Only complete sides of disks will be copied.
e) Instructions on how the disks are to be returned, i.e., at the next meeting, by mail, etc. (IF RETURN IS TO BE BY MAIL PLEASE BE SURE TO INCLUDE A STAMPED, SELF-ADDRESSED FLOPPY DISK MAILER, AND $2.50 TO COVER SHIPPING COSTS).

DISK OF THE MONTH-

Each month a collection of some of the best programs donated to APPLE PI during the preceding month are made available. The DOM can be purchased at any group meeting or by mail.

DUES-

Current membership dues are $18.00 per year, with a one-time application fee of $10.00. Checks should be made payable to APPLE PI.

...Continued from 1

```
10  NUMOFCELLS = NUMOFCELLS - 1
15  FLAG = 1
20  FOR L = 1 TO NUMOFCELLS
30  IF  D(L) > D(L+1)  THEN
      T = D(L) :
      D(L) = D(L+1) :
      D(L+1) = T :
      FLAG = 0
40  NEXT L
50  IF FLAG = 0 THEN 10
100 END
```

Listing 3. Andre's sink sort.

Now we are only comparing NUMOFCELLS items half as many times! Thus 100 iters are checked 50 times for about 5000 comparisons. This is a great increase in efficiency, but I found that this is not quite the case. In fact the time is only 33% better than the bubble sort. I wondered about this and thought I might know what was wrong.

Basically what I noticed was that in line 50 we are using a GOTO statement. Because Applesoft starts at the beggining of the program and looks down the list at each program line until it finds the one for the GOTO this can slow our sort down considerably if the routine is at the middle or end of a very long program. A FOR..NEXT loop is much more efficient and also gives us the advantage of not changing the actual value of NUMOFCELLS:

```
10  FOR I = NUMOFCELLS-1 TO 1 STEP -1
15  FLAG = 1
20  FOR L = 1 TO I
30  IF  D(L) > D(L+1)  THEN
      T = D(L) :
      D(L) = D(L+1) :
      D(L+1) = T :
      FLAG = 0
40  NEXT L
50  IF FLAG = 1 THEN I = NUMOFCELLS
60  NEXT I
100 END
```

Listing 4. Improved Sink sort.

This routine does the same thing that Andre's does it just does it a little more efficiently. We are still stuck with fantastic numbers of comparisons if our list is long, and I had not found out the reason that the routine took only 33% less time than the bubble sort.

In his article Andre' mentions that he believed that someone else must have came up with this same idea before but he could find no published results. I wondered about this myself and investigated a bit. What I discovered was that the improved sink sort was very nearly dentical to another sort method I had developed using different reasoning. In order to understand how I came about my sorting method let us first take a closer look at the bubble sort algorithm.

52 Card Pickup...

Let us imagine that we have a deck of cards each marked with a different number from 1 to 52. Suppose we shuffle them and then lay them out in a line so that we can see all of them. Using the bubble sort we would look at cards 1 and 2 and if they are out of order we would switch them. Then we would look at cards 2 and 3 and if they are out of order switch them. We do this until we reach cards 51 and 52 then we go back to the begining start over again, repeating the process 52 times. It occured to me that it was a bit narrow minded to be looking only at two cards next to each other. If we were to devise a method we would probably look down the entire line until we found the smallest number, place it at the beggining and then look for the next card. In a sort routine we can not assume that the number 1 will be the first number so we must take a slightly different approach. My routine picks up the first card then looks down the line for a card with a smaller value than this one. If we find one we swap them. If not we put it back down and pick up the second card and go down the line. We do this once through the list until we are finished. After that the cards are sorted. Here is how it would look in Applesoft:

```
10  FOR I = 1 TO NUMOFCELLS-1
20  FOR L = I+1 TO NUMOFCELLS
30  IF D(I) > D(L) THEN
      T    = D(I) :
      D(I) = D(L) :
      D(L) = T
40  NEXT L
50  NEXT I
100 END
```

Listing 5. Steve's search sort.

If we compare Listings 4 and 5 we discover one difference is that in one we start at the top of the list and work down but in the other we start at the bottom and work up. So the counting method works out the same and we end up saving half the time of a bubble sort. When I tested it I found out that, indeed, my sort does save 50% of the time of a bubble sort. What is the difference? Upon closer comparison we notice that in the bubble sort, as well as Andre's sink sort, we are constantly making switches everytime we find to elements next to each other that are not in order. In my sort we make a switch with an element that is smaller than the first, then we will have to find an element that is even smaller if we are to make another switch. This means we actually make fewer switches which takes less time.

There is one, minor, disadvantage to my sort. Since we are looking over the larger distances we can not assume that if no swap was made the list is sorted. Thus we can not use a FLAG and we might lose some speed. I find my method a little more intuitive and thus I prefer it when sorting short lists. Long lists still remain a problem. To greatly reduce the sort time for large arrays we will need to look at some radically different sort methods.

Shell sort...

The first improved sorting method we should look at is known as the Shell sort, or very typically the Shell-Metzner sort. Going back to the expamle of the cards what the shell sort does is first go through comparing cards in the first half with cards in the second half of the deck. So we would compare card 1 with card 27, card 2 with card 28 and so on. If we find two that are not in the right order we swap and proceed. Then we take half the distance and do it again. So card 1 is compared to 14, card 2 to 15 and up till card 39 is compared to 52. Then we halve the distance again. We keep doing this until we are looking at the cards right next to each other as in a bubble sort. Once this last pass is made the cards are sorted. Here is the shell sort in Applesoft:

```
10  JUMP = NUMOFCELLS
20  JUMP = INT(JUMP/2)
30  FLAG = 1
40  FOR I = 1 TO NUMOFCELLS - JUMP
50  L = I + JUMP
60  IF D(I) > D(L) THEN
        T = D(I) :
        D(I) = D(L) :
        D(L) = T :
        FLAG = 0
70  NEXT I
80  IF FLAG = 0 THEN 30
90  IF JUMP > 1 THEN 20
100 END
```

Listing 6. Shell-Metzner sort.

The INT command in line 20 is necessary or else the last element in the array may not get sorted. It is harder to figure out how many comparisons are needed by this sort. Basically I have figured out that it is about NUMOFCELLS/2 items done NUMOFCELLS/2 times or one fourth the number needed by the bubble sort of listing 1. A comparison of all the sort routines will be given at the end of the article.

Quick Sort...

This last sort routine uses what I believe to be the least intuitive method, but it gives by far the fastest results. Going back to our deck of cards what the quick sort does is take an arbitrary card somewhere in the middle of the deck and looks for one on the left that is bigger and one on the right that is smaller. Once two such cards are found they are swapped. This is continued until all numbers on the left are smaller than all numbers on the right. Then the two halves are divided, a random number from each is selected and the process repeated. This is done until there are as many divisions as there are cards, at that point the cards are sorted. The number of times that the cards must be gone through is proportional to the LOG of the numbers of cards. For 100 elements bubble sort must go through 10000 comparisons and quick sort takes about 1000. Here is the Applesoft version of quick sort:

```
10  STACK = 1
20  LEFT(1) = 1
30  RIGHT(1) = NUMOFCELLS
40  LEFT = LEFT(STACK)
50  RIGHT= RIGHT(STACK)
60  STACK= STACK-1
70  L = LEFT
80  R = RIGHT
90  X = D((L+R)/2)
100 IF D(L) < X THEN
        L = L+1 :
        GOTO 100
110 IF X < D(J) THEN
        R = R-1 :
        GOTO 110
120 IF L > R THEN 150
130 T = D(L) :
    D(L) = D(R) :
    D(R) = T
140 L = L+1 : R = R-1
150 IF L <= R THEN 100
160 IF L < RIGHT THEN
      STACK = STACK+1 :
      LEFT(STACK) = L :
      RIGHT(STACK) = R
170 RIGHT = R
180 IF LEFT < RIGHT THEN 70
190 IF STACK<>0 THEN 40
200 END
```

Listing 7. Quick sort.

Wasn't that easy! The two arrays LEFT() and RIGHT() were neccesary for keeping track of the divisions we will need for the sorting. These arrays should be dimensioned to a bit more than the log of the main arrays dimension, thus if DIM D(1000) then LEFT() and RIGHT() should be diminsioned to about 20. Quick sort suffers from not being able to tell if the array is sorted already so it will always take about as long to sort any list.

GOTO 10

JOHNNY APPLESEED

Dear John,

I have been working on a Applesoft Program for quite some time now, and, as with all lengthy Applesoft Programs, I have begun to have problems with memory. I have a 64K Apple, but I don't know how to use the "extra" 16K. Can you help?

Out of Memory

Dear Out of Memory:

Your problem is a familiar one. Many Apple users have 64K machines, but they are unable to use them because of the fact that normal Applesoft doesn't recognize the top 16K bank of memory. There is, however, one way to get around this.

Some of the DOS alternatives (like Diversa-DOS) have utilities on the "seed" disk that allows you to move DOS up to the top 16K bank of memory. By moving DOS up to the top 16K bank, you have the 10K that DOS normally takes up all to yourself.

John

Dear John,

I have a copy of ProDOS that I got with my Apple //e. The other day we tried to run it on a friends Franklin Ace. We could not get it to run. I thought that ProDOS would run on Apple II, II+ and the look a likes. Am I doing something wrong?

Pro-plexed

Dear Pro-plexed,

Apple has designed ProDOS to run on Apples. It looks at the hardware and if it is not an Apple it croaks. Apple is doing a lot of this sort of thing with new Apple products. Someday you may have a serial number in ROM that will be a key to running your Apple software. If you try to run on any other computer, including other Apples, the software will not work. This is to keep one program on one computer. This is now being done on the Lisa. I'm sure that it won't be long before some third party software house develops a ProDOS look alike that will run on the other systems.

John

COMPUTER PRINTERS

BROTHER HR-15

This compact printer is ideal for reports and word processing. The optional keyboard allows you to use it as a correctable typewriter—and it's easy on your budget.

Orange Micro offers the lowest prices, the finest selection, and the interfacing expertise needed to make a printer work with your system.

STAR MICRONICS GEMINI 15X
If you're working with a limited budget, the Gemini 15X is your business printer. Now you can print your ledgers and spread sheets in an easy to read wide carriage format.

The finest brand names available:
• AMDEK • ANADEX • BROTHER • DATAPRODUCTS
• EPSON • GRAPPLER + • IDS • MEMOREX • NEC
• OKIDATA • RANA • STAR MICRONICS • TELEVIDEO

Orange Micro PRINTER STORE
"Computer Printer Specialists"

Call for a complimentary Printer Comparison Chart
Showroom Hours:
Mon.–Fri. 10–6
Sat. 10–5

MasterCard VISA AMERICAN EXPRESS
No Personal Checks
Telephone Sales Welcome

AURORA
1941 S. Havana

(303) 750-5900
Havana Square

© Orange Micro, Inc., 1983

Graphics & ePlot

by Andre' van Meulebrouck

I predict graphics will play a bigger role in the future of computers than all but the most fanatical of graphics fans could ever imagine.

Graphics are already playing a role in the following; computer aided design for numerous fields including logic circuits, drafting, illustrations, plotting mathematical functions, alternate character sets, logos and mimicking type-setting to name a few.

Of course, graphics are most famous for the role they play in video games which I think is unfortunate because as a result of video games, many people think graphics have no value other than for animation in games -- like the thing that ate the blob, or the blob that ate the thing -- whatever. (If I stepped on any toes, you can leave me hate mail on the Pi BBS -- #156).

Lately, another use of graphics has been highlighted with the advent of the Macintosh. Like the Lisa, it creates a graphics environment for the user that is very user friendly. More importantly, its display is completely bit-mapped. Bit-mapped means mapping out the screen in memory -- there is a memory image that corresponds to, and decides the screen display. Certain memory cells control certain pixels on the screen. To change the screen display, the appropriate memory cells are manipulated. When you declare a graphics mode on an Apple II (and most other computers), you are then in a bit-mapped mode. If you are using Applesoft, it is taking care of the low level specifics for you so all the bit-mapping is invisible to you. Text modes on most computers are not bit-mapped. Video display hardware/firmware is taking care of the specifics of how text is formed in terms of pixels, and no simultaneous memory image of the screen is kept (at least not as graphics information). The video display generator illuminates the appropriate pixels for each ASCII code it recieves. On a Macintosh or Lisa, even text is bit-mapped, so in essence, you are "permanently" in graphics mode.

What does bit-mapping buy? Well, for one, mixing text and graphics is no problem! You can "cut and paste" text and illustrations together in the same document with ease. Another advantage of bit-mapping is that you can mix numerous fonts and user created characters in the same document, as well as going in and changing the shapes and sizes of the characters the computer put on the screen. Furthermore, what you see is what you get -- the printouts of bit-mapped graphics are a snapshot of the screen. That is why Lisa-technology is such a revolutionary change in the micro-computer world. Of course, a computer like a Macintosh makes special demands on printers. Printers are mode sensitive just like most computers are about graphics and text. Printing bit-mapped displays requires a printer to be "permanently" in graphics mode. Apple has put out just such a printer to go with the Macintosh (the Imagewriter).

Still, I don't think the //e is going to be eating humble pie because of the Macintosh. There is a trend developing among Apple II owners -- many are selling their II's and buying Macintoshes, even though the amount of Macintosh software is not what it is for II's (not yet anyway). I think there are great products yet to come for the //e -- the expansion slots will allow the //e to grow with the future. (The mouse is already out for one).

GOTO 8

SECRETARY'S REPORT

(April Meeting)

The Meeting was convened by Dave Eisler at 7:00 pm, April 5, 1984, at the Greene Center, Colorado School of Mines.

He announced that next month APPLE Computer would present a "Surprise" program. New Officer elections will also be held at the May Meeting. The Nominating Committee, headed by Tom Angelos, now has at least one candidate for each office. Nominations for additional candidates were requested from the floor, and Tom announced that he could be contacted by members wishing to nominate themselves or others up until April 12th. The current roster of nominees are:

President:	Bob Hein
Vice President:	A. Van Meulebrock
Secretary:	Jim Lyons
Treasurer:	Will Smith
Directors at Large:	
(Choose three)	Trent Hein
	Bill Hatchett
	Joseph Negy
	John Worthington
	Tom Decker
	Bill Cronister
	Randy Ansley

Other announcements included:

-- Copies of HOME COMPUTER MAGAZINE were distributed free to the members present. The Magazine provided over 200 copies of their current issue.

-- The April SWAP Meet will take the format of a Garage Sale. Members are invited to bring equipment which they would like to sell.

-- Dave Eisler is offering a Machine Language Course. Contact him if you are interested in taking it.

-- Dan Ryan, a new MacIntosh owner, is interested in starting a group with other Mac owners. Call him at 424-8197.

-- Clyde Pearce, a member who recently had his Apple II+ stolen, had some hard earned advice to share with the membership. Keep your serial numbers and receipts somewhere safe and separate. Check your insurance policy to make sure it will fully replace the items lost, so that you won't wind up with the insurance people telling you that your machine is now so obsolete and has depreciated so much "that you owe them money for it!"

Trent Hein demonstrated the Disk of the Month, and announced a 9:00 deadline on the "Most Valuable Program Contest".

The door prizes were a copy of PRODOS, donated by ADAMS COMPUTERS and won by Dan L. Stieren, and a Verbatim Datalife Holiday Pack, donated by Apple Pi and won by William Green.

ADAMS COMPUTERS also manned our Guest Sales Counter.

The main program was on Printers and Printer Interfaces presented by Bob Brossman and Jerry Arnold of Orange Micro (1941 S. Havana, Aurora, Co)

The meeting was adjourned at 9:00 pm.

LIBRARIAN'S REPORT

The April Swap meet marked the end of our M.V.P. Contest. We had a total of five entries. Results on the winning entries are shown elsewhere in this issue of the Seed.

As already announced, the May DOM will consist of all the entries into the M.V.P. Contest, plus a few "extras".

We really need some new public domain CP/M and Pascal programs for our CP/M and Pascal DOMs. If you have any you would like to contribute, please call me or talk to me at one of the meetings. A public domain software library only works when you do!!!

Finally, I want to thank Dave Eisler for contributing the "Backwards" print subroutine for the April DOM. I think it was a super "April Fool"!

----> Trent Hein, Librarian <----

...Continued from 6

For those who own a //e with a revision "B" motherboard, and an extended 80 column card, there is a very nice software package out that will allow you to take advantage of the //e's capability to do double hi-res graphics. Hi-res on an Apple II allows you 280 horizontal pixels by 192 vertical pixels. The //e's double hi-res screen gives you more horizontal resolution, but no more vertical resolution: 560 horizontal pixels by 192 vertical. (Note: none of the Apple II's have square pixels like the Macintosh does.). Some magazines have carried articles describing how to do double hi-res graphics in Assembly Language. I've never met anyone who has actually tried it, but everyone I know of who has the right equipment would like to do double hi-res graphics. Well, thanks to ePlot, you don't have to be an Assembly programmer to do double hi-res stuff. A company called "Microseeds" has come out with a set of routines that give you an extension to Applesoft to allow you double hi-res commands. These routines are ampersand routines, and behave as ampersand routines do. The commands that are added look like the usual hi-res commands, only with an ampersand in front (i.e. &HGR, &HPLOT, &HCOLOR). To use ePlot, you boot the ePlot disk. After putting the ePlot routine into memory, you load or write your program as usual, making use of the added commands as you like. Normally, when you get more resolution, you begin losing color(s). Not so with ePlot! It allows you 16 colors! (Contrast that with the normal Apple graphics modes -- 16 colors in low-res, 6 colors in hi-res! Also, contrast that with the Macintosh -- monochrome only, 512 horizontal pixels by 342 vertical.). The disk has some sample programs and other routines you might find useful.

Vendor information -- ePlot is a product of: Microseeds, 20 Goodell Road, Stafford Springs, CT 06076. (203) 684-4631. I purchased ePlot for $29.95 in November, 1983 from: JQB Enterprises Inc., P.O. Box 269 -- 10 Fletcher St., Ayer, MA 01432. Phone (617) 647-0719. Toll free order phone (ask for operator # 592): 800-824-7888 Cont. US, 800-824-7919 AK, HI, 916-929-9091 Worldwide.

Thank you!

ADAM'S COMPUTERS INC. 3/85
3355 S. WADSWORTH BLVD.
LAKEWOOD CO 80227
6 303-986-9655

COLORADO COMPUTER SYSTEMS 1/85
3005 W. 74TH AVE.
WESTMINSTER CO 80030
5 303-426-5880

COMPUTERLAND OF DENVER
2422 S. COLORADO BLVD. 12/84
DENVER CO 80222
1 303-759-4685

COMPUTERLAND OF N.DENVER 1/85
8749 WADSWORTH BLVD.
ARVADA CO 80003
4 303-420-1877

NEIGHBORHOOD COMPUTERS
13045 W. ALAMEDA PARKWAY 1/85
LAKEWOOD CO 80228
3 303-988-9140

LITTLE PEOPLE AND THE IIe=LOVE !

COMPUTER SOFTWARE DRAWING PROGRAMS ARE DEFINATELY NOT JUST FOR DRAWING!

Our Day Care uses drawing programs to teach a few of the following concepts, very effectively;

The "I control the computer" psycology
Computer language
Right-left-up-down
North-south-east-west
Letter recognition
Logical thinking
Planning
Number concepts
Pre-programming learning
**Keyboard familiarty

**(We favor keyboard controlled drawing programs over Koala-type pads and joystick control to promote keyboard familiarty.)

Two of the five drawing programs in constant use in our daycare, in order of difficulty level are;

EARLY GAMES FOR YOUNG CHILDREN (ages 2 - 6)

Basic and simple enough for a two year old. Line drawing is controlled by different sections of the keyboard; i.e. any key on the left side of the keyboard moves the line left; anything on the top row (numbers) moves the line upward. Hitting the space bar changes the color of the line. The lines are capable of moving up, down, left, right, even diagonally. Pictures can be named and saved for future comparisions and additions.
EARLY GAMES FOR YOUNG CHILDREN also includes several other excellant pre-school letter-number-shape activities.

SPINNAKER KINDERCOMP'S "DRAW" (ages 5 - 8)

Also included with some fun, easily handled learning games, (in fact, this is one of the few early learning programs that include lower case letter recognition with capital letters) Kindercomp's Draw programs line movement is controlled by using four letter keys, I, J, K, and M. C is used to change line color, and F is used to fill in color in an area. (It is also possible to do joystick drawing with this program) We find only two minor drawbacks to this program - in our day care we like to give the little people responsibilitys that lead to respect for the equipment; they boot their own disks, change menues etc. Naturally that is a little risky because of their ages (3- 9), so back-ups of constantly used programs are essential - we have written Spinnaker several times to request backups with out sending in damaged disks, but to no avail. Also the pictures created on Kindercomp cannot be saved.

NEXT COLUMN; MICROZINE'S "PAINT
SPINNAKER'S "DELTA DRAWING"
AND LAST, BUT CERTAINLY NOT LEAST-THE GREAT TERRIPAN LOGO!

If you have any comments or questions, please feel free to call: Liane Nusse 279-5763.

...Continued from 4

Speed...

I have done benchmark tests with all of these sorts and came up with the following time comparisons: For 100 Elements:

Bubble sort:	120 seconds
Sink sort:	80 seconds
Search sort:	60 seconds
Shell sort:	30 seconds
Quick sort:	12 seconds

If pure extrapolation were correct (and it may not be) then 1000 elements would take over 3 hours with bubble sort. I have checked it and quick sort takes just under 3 minutes on 1000 elements!

Helpful advice...

I don't want to leave anyone with the impression that quick sort should be used all of the time. In fact it should not be used unless you have very long arrays or arrays that are always messed up. Bubble sort is sufficient for arrays that have less than 20 elements, or under 30 elements if they are usually already in order. The sink sort would be about right for anything less than 50 elements or so. Search sort is good up to about 100 elements, more if you are patient as I am. I would recommend the shell sort for arrays up to about 200 elements. Over that size quick sort can not be beat. Remember that size is not the only consideration. More program lines mean more memory space and things start to really add up. Don't waste time and effort using a fancy sort routine for alphabetizing the names of a dozen dinner guests!

Note to PASCAL users...

One last thing for those of you who use PASCAL. Quicksort was originally designed to make use of recursion. Did you notice that instead of dividing up the array with partitions we could have just told the sort routine to sort the smaller divisions, then sort those, and so on? This is the very thing recursion is supposed to help with. Here is quick sort for PASCAL:

```
procedure SORT (LEFT,RIGHT :
                          integer);

  var
    I,J,X,T  : integer;

  begin
    I := LEFT;
    J := RIGHT;
    X := A[(LEFT+RIGHT) DIV 2];
    repeat
      while A[I] < X do I := I+1;
      while A[J] > X do J := J-1;
        if I <= J then
          begin
            T := A[I];
            A[I] := A[J];
            A[J] := T;
            I := I+1;
            J := J-1;
          end;
    until I > J;
    if I < J then SORT(L,J);
    if I < R then SORT(I,R);
  end.
```

Listing B. Recursive quick sort.

Well that about wraps it up. I hope that this is helpful to new programmers trying to understand sorting methods. For the oldtimers, perhaps I have uncovered something new you can make use of.

---

follow up

As a follow-up to the recent review of the Spies Super MX interface card, I have a twelve page "manual" written for Screenwriter II, the Spies card, and the Epson FX 80. This was prepared for my own use. Includes an example of a special character set downloaded into the Epson's RAM, lots of examples of Epson control and escape commands "translated" into Spies format.

Send a self-addressed stamped ($0.54) envelope and I'll pass it on. Letter size will work but 9" x 12" is better.

A purchaser should be sure to get version 1.06; prior versions did not space lines per inch correctly.

Also would like to unload some "one-owner" games, mostly rp and adv at about 1/4 of list.

John Kellogg
91 Fairway Lane
Littleton CO 80123

TURBO PASCAL & CP/M

by Dennis Lunt

Pascal is becoming a programming language that is transcendng the boundaries of operating systems. TURBO Pascal, a product of Borland Iternational, is a language system which may be used on Apples which have CP/Mcards. Mine is an Apple /// with a SoftCard from Microsoft. This article is to give instructions for installing the screen editor on the Apple,a necessary function which for some reason is not supported by the orignal Borland International program.

The installation begins by running the TINST routine on the program disk. S)creen Installation is then selected from the first menu. "Choose One Of The Following Terminals:" and a list of various computers and terminals appears on the screen. An obvious omission from the list of 25+ sytems is "Apple". One is left with choice 30) None of the above. Following te sequence of events as described in Appendix K of the user's manual, onemust provide TURBO Pascal with Apple's control parameters. Having had a request for such information from Borland International unanswered, and subsequently determining the proper parameters on my own, the following instruction set is provided for other interested programmers:

Do you want to modify the definition before installtion? answer: "Y"

Terminal type: answer: "Apple"

Send an initialization string to the terminal? answer: "N"

Send a reset string to the terminal? answer: "N"

Cursor Lead-In command: answer: "^Z" (Control Z)

Cursor Positioning command to send between line and column: answer: (none)

Cursor Positioning command to send after line and column: answer: (none)

Column first? answer: "Y"

Offset to add to line: answer: "0"

Offset to add to column: answer: "0"

Binary address? answer: "Y"

Clear Screen command: answer: "^\" (Control \)

Does Clear Screen also HOME cursor? answer: "Y"

Delete Line command: answer: "^Y" (Control Y)

Insert Line command: answer: "^N" (Control N)

Erase To End Of Line command: answer: "^_" (Control Shift -)

Start Highlighting command: answer: (none)

Number of rows (lines) on your screen: answer: "24"

Number of columns on your screen: answer: "80"

Delay after Cursor Address (0-255ms): answer: "0"

Delay after Clear, Delete, and Insert (0-255ms): answer:"0"

Delay after Erase To End Of Line and Highlight On/Of (0-255ms): answer: "0"

GOTO 14

PRES. $TRINGS

I'll be returning to the real world next month, after spending two years as an officer of Apple Pi. But before going, I want to thank all the terrific people who have been doing all the work. In particular, I want to thank Randy Ansly, Bob Hein, and Dave Nelson. These are the three guys who made the club go last year. Month after month, they were always there, always doing a great job, always doing ten times more than they had to. When times got tough, when other people didn't show up, these guys filled in, took on extra work, pushed projects thru, and kept the club running. The entire membership owes them a great deal.

I owe Dave for all the times he helped me organize and remember things. I owe Bob for all the advise and motivation. And of course, I owe Randy for making it all fun. Next year I might miss working with them.

This is a good time to reflect on last years accomplishments. Let's thank all the wonderful people who contributed their sweat and time to make them happen.

started the newcomers group
put together a bulletin board
created store benefit package
computerized the door prizes
improved the Seed
christmas party
started class bulletin board
implemented a budget
put taxes in order
revised the bylaws
garage sale
most valuable program contest

It was alot of fun, and alot of work too. Here's to a great term, and an even better one next year!

- Dave

TREASURER'S REPORT

Beginning Balance (3/1/84)		$2297.30
INCOME		
New Memberships		$ 756.00
Renewal Memberships		378.00
DOM Sales		335.00
Library Copies		58.00
Sales of Supplies		2891.00
Interest Earned		25.33
SEED Advertising		60.00
Store Membership		100.00
	TOTAL	$ 4603.33
EXPENSES		
Supplies for Resale		$ 2790.50
CSM Room Rental		759.00
Librarian Supplies		155.50
Printing Costs		635.00
Postage		171.70
Supplies		107.34
	TOTAL	$ 4610.04
LIABILITIES		
Sales Tax Accrual (3 months)		361.12
	TOTAL	$ 361.12
ASSETS		
Supplies for Resale a/o 3/29/84		$6513.70
Cash		2370.57
CSM Room Rental (thru June)		420.00
	TOTAL	$9304.27

**** MVP CONTEST RESULTS ****

1ST PLACE: RANDY ANSLEY
2ND PLACE: RON HEATH
3RD PLACE: G.C. GRAY

ENTRIES WILL APPEAR ON THE
MAY 1983 DISK-OF-THE-MONTH

****

(REM: Reprinted from The MAPLE ORCHARD, Vol. 3, no. 2)

SAVING APPLE LOGO GRAPHICS

Wayne Stuart

"APPLE LOGO", from Logo Computer Systems has many capabilities. However, I use the language primarily to explore Turtle Geometry and to draw pictures. This article tells how users of "APPLE LOGO" can save their own LOGO pictures for later use.

To draw a picture on the computer screen, the LOGO user types in words such as 'FORWARD', or 'LEFT'. These commands constitute the program for the LOGO 'turtle'. Following its programmed instructions, the turtle moves across the screen, tracing out a picture as it goes.

The following sample LOGO program is named "HEXAGON". It draws a six-sided figure like those hidden in the illustration, opposite. In fact, the "PARQUETRY" program uses this three line program as a sub-routine.

```
TO HEXAGON
REPEAT 6 [FORWARD 30 RIGHT 60 ]
END
```

The LOGO command ' .PRINTER 1 ' lets users print out such LOGO programs on paper. (Assuming that the printer card is installed in the Apple's slot #1)

"APPLE LOGO" also lets users save programs as text instructions on a disc. The LOGO command to do this is: ' SAVE "your program name '.

To redraw an earlier picture on the screen, I could simply load the text instructions from the disc, and then type the name of the program. The LOGO turtle then executes the program, redrawing the picture as before.

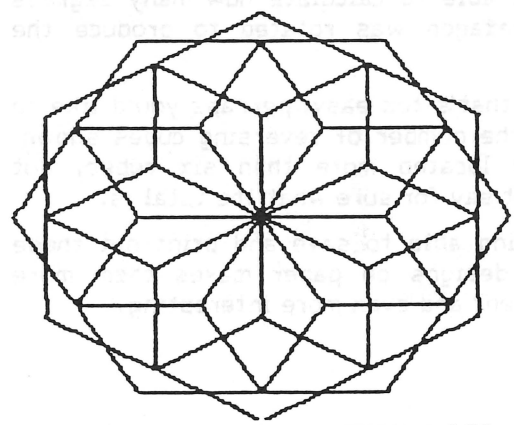
Logo Computer Systems supplies two extensive books with the language to explain its operation. One is the "Apple Logo Reference Manual", the other "Apple Logo: Introduction to Programming through Turtle Graphics."

Neither of these books, however, explains how to save the actual picture as a binary file, nor do they mention how to print out a picture as it appears on the screen.

The Krell and Terrapin versions of LOGO have included provisions for saving pictures on disc, and for printing pictures onto paper. "APPLE LOGO" lacked such provisions, or so I believed.

Wanting to print out LOGO pictures, I sought advice from salesmen in local computer stores but they couldn't help me. I finally found the key in an article by Steve Birchall. ("Softside", Vol. 6, No. 3, p.87).

In reviewing the book, "Apple Logo", by Harold Abelson, (Byte Pub., 1982.), Birchall mentions that "printouts of Turtle Graphics" are overlooked. He then explains how to do it, giving as his source, "POLYSPIRAL", the newsletter of the Boston Computer Society's LOGO User's Group. The procedure is explained here for the benefit of "Maple Orchard" readers who may not be able to obtain the original article.



"PARQUETRY" -- FROM LOGO

Let's assume we want to save such a picture as a binary file on a diskette. Having created your picture on the LOGO screen, you must then leave "APPLE LOGO" and switch to "BASIC".

To do this, replace the LOGO disc in the drive unit with an initialized BASIC diskette.

The LOGO command ' .PRINTER 6 ' boots the BASIC disc. In so doing, it substitutes BASIC for LOGO in the Apple, but it keeps the picture intact in the computer's memory.

To save your picture on the disc, type "BSAVE your picture's name ,A$2000,L$2000."

Having done this, you can use a BASIC or machine language program to retrieve the picture from the disc in the usual way.

One major advantage of storing pictures as binary disc files, is that they can be retrieved and printed out on paper. To do this, use a suitable graphics screen dump program and printer. This article and picture, for example, were both originally printed out using the "Gutenberg Word Processor" which prints pictures as well as text.

The "PARQUETRY" design was created by a LOGO program. The program drew a hexagon, rotated it, and redrew it several times to create the illusion of interlocking cubes.

Now that you know how to save and print out such pictures, how about trying to apply some Turtle Geometry of your own? By examining the illustration more closely, you may be able to calculate how many degrees each hexagon was rotated to produce the figure.

If that's too easy, perhaps you'd like to count the number of reversing cubes shown. I have located more than six cubes, but couldn't say for sure what the total is.

Being able to save and print out these LOGO designs on paper makes them more permanent and even more interesting.

```
20 T$ = "CRAE : CO-RESIDENT APPLE
     SOFT EDITOR": HTAB 21 -  LEN
     (T$) / 2: PRINT T$: HTAB 19:
      PRINT "AND"
30 T$ = "APA : APPLESOFT PROGRAMM
     ERS ASSISTANT": HTAB 21 -  LEN
     (T$) / 2: PRINT T$
40  VTAB 6: PRINT "INSERT A DISK
     CONTAINING THE 'CRAE' FILE";
     : PRINT : HTAB 15: INVERSE :
      PRINT "EDIT.PROD": NORMAL
50  PRINT : HTAB 5: PRINT "IN THE
      DRIVE AND HIT A KEY  ";: GET
     A$: PRINT
60  PRINT : HTAB 9: PRINT "LOADIN
     G  CRAE 2.0"
70  PRINT  CHR$ (4);"BLOAD EDIT.P
     ROD": PRINT
80  PRINT "INSERT DISK WITH 'RBOO
     T' AND 'APA' AND"
90  HTAB 7: PRINT "HIT ";: INVERSE
     : PRINT "SPACE BAR";: NORMAL
     : PRINT " TO CONTINUE ";: GET
     A$: PRINT
100  POKE 1013,76: POKE 1014,00: POKE
     1015,141: POKE 111,240: POKE
     115,240: POKE 112,121: POKE
     116,121
110  PRINT : PRINT "LOADING APPLE
     SOFT PROGRAMMERS ASSISTANT"
120 ADRS = 0: PRINT  CHR$ (4),"BL
     OAD RBOOT": CALL 520
130 ADRS =  USR (0),"APA": FOR I =
     0 TO 2:P(I) =  PEEK (1013 +
     I): NEXT : IF P(1) = 0 THEN
     P(2) = P(2) - 1:P(1) = 256
140  POKE ADRS + 301,P(1) - 1: POKE
     ADRS + 302,P(2)
150  DATA 67,82,65,69,32,32,32,32
     : FOR I = 0 TO 7: READ A: POKE
     ADRS + I + 190,A: NEXT : POKE
     ADRS + 190 + I,0: CALL ADRS:
      END
```

...Continued from 11

```
     Is  this  definition  correct?
     answer: "Y"

     Operating  frequency  of  your
     microprocessor  in  MHz  (or
     delays): answer: "4"
```

The installation is thus performed, and the editor should work properly. If your system is other than the SoftCard, then things like the microprocessor frequency may be different. Check the manufacturer's specs.

That's your start, Pascal programmers. Blaise On!

- S O F T W A R E R E N T A L S -

Try it before you buy it!

1/2 of any rental can be applied to the purchase

DISCOUNT PRICES are available on all purchases

Used software sold on consignment

CHECK OUT these every day LOW PRICES

SINGLE SIDED / DOUBLE DENSITY DISKETTES - BOX OF 10.....$ 19.95

BLACK EPSON MX70, MX80, FX80, RX80 PRINTER RIBBONS.....$ 5.50

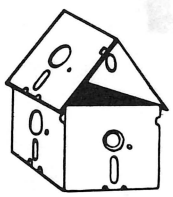
LASER PERF PAPER - 20lb. - BOX OF 2600 SHEETS.....$ 24.95

MAILING LABELS - 1 ACROSS - 5000 LABELS.....$ 19.95

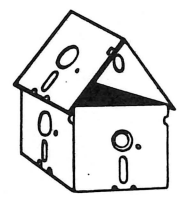
70 DISKETTE FLIP FILES.....$ 19.95

Join the software rental club and get 15% off the above prices.

- C A L L F O R D E T A I L S -



BOULDER SOFTWARE HOUSE
Table Mesa Shopping Center
Boulder, CO 80303
(303) 499-3456



DISK OF THE MONTH ORDER FORM

This form is for those members who are unable to attend the next meeting of Apple Pi and would like to purchase a DOM via mail. DOMs will still be available at the monthly meeting and swap meet. It is also possible to order past DOMs.

CIRCLE THE APPROPRIATE DOMs THAT YOU ARE ORDERING

JAN FEB MAR APR MAY JUN JUL AUG SEP OCT NOV DEC

1980 1981 1982 1983 1984 1985

[ ] DOS 3.3 [ ] Pascal [ ] CP/M

QUANTITY	ITEM	UNIT PRICE	TOTAL
_____	DOMs marked above	$5.00 ea	_____
	Postage and Floppy Mailer	$1.25 ea	1.25
		TOTAL	_____

NAME_____

ADDRESS_____ CITY,ST_____

ZIP_____ PHONE _____

Send form to Apple Pi, P.O. Box 17467, Denver, CO 80217-0467

VOLUME 6 NUMBER 5 MAY 1984

denver apple pi Seed

newsletter of the apple pi computer users' club

P.O. Box 17467
Denver, CO 80217

Price $1.50

BULK RATE
U.S. POSTAGE
PAID
DENVER, COLO.
PERMIT NO. 3182

DID YOU REMEMBER TO VOTE?